POSIX Support for the DII COE Kernel

Realtime and Embedded Systems Forum

The Open Group

Austin 2001

Presenter: Doug Robinson
Systems Engineer -
DII COE Technology
Insertion Task

Agenda

- Brief discussion of the history of the DII COE
- Discuss the role of POSIX by the DII COE
- Lessons learned from the evolution of DII COE
- Future kernel directions

DII COE: Defense Information Infrastructure Common Operating Environment

DEFENSE INFORMATION SYSTEMS AGENCY

JPL

DII COE History

Goals of the DII COE

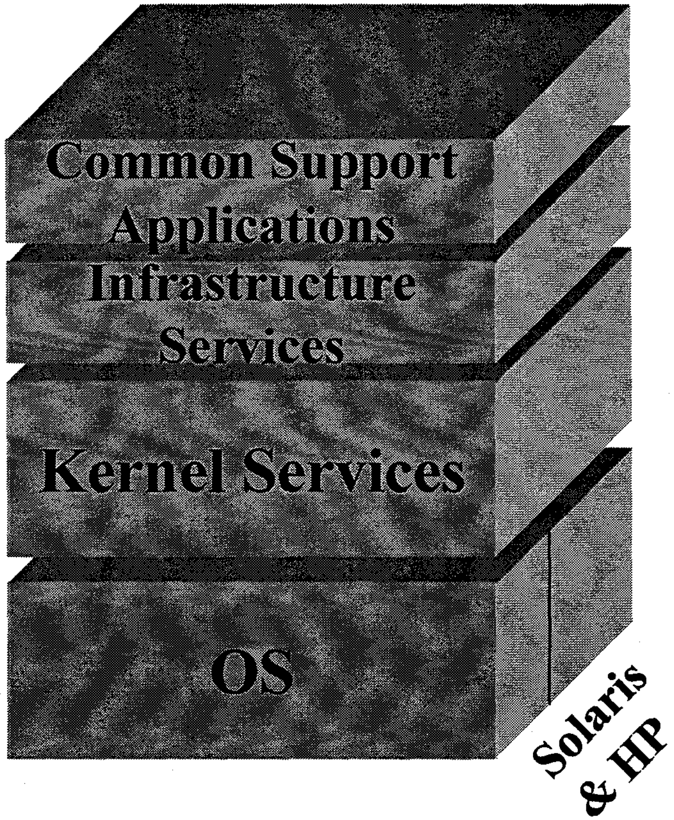
- Provide an abstract platform that isolates programmers from OS and hardware implementations
- Provide a common method for administering DII COE systems - must be easy to use and economical
  - Account management
  - Software installation
  - Peripheral management (disks, printers, etc.)
  - Security enhancements
- Provide a mechanism by which hardware vendors can self-certify their platforms
  - Kernel compliance test suites

DII COE Architecture

- The DII COE utilizes a layered model:
  - Mission Applications
    - Task-specific applications, unique to an individual assignment
  - Common Support Applications
    - Common user application that are applicable to a large number of assignments
  - Infrastructure Services
    - Toolkits/services that are utilized by common support and mission applications
  - Kernel
    - Kernel Services
      - Common APIs that isolate OS-specific calls from the infrastructure and higher services
    - Operating System
      - Vendor supplied

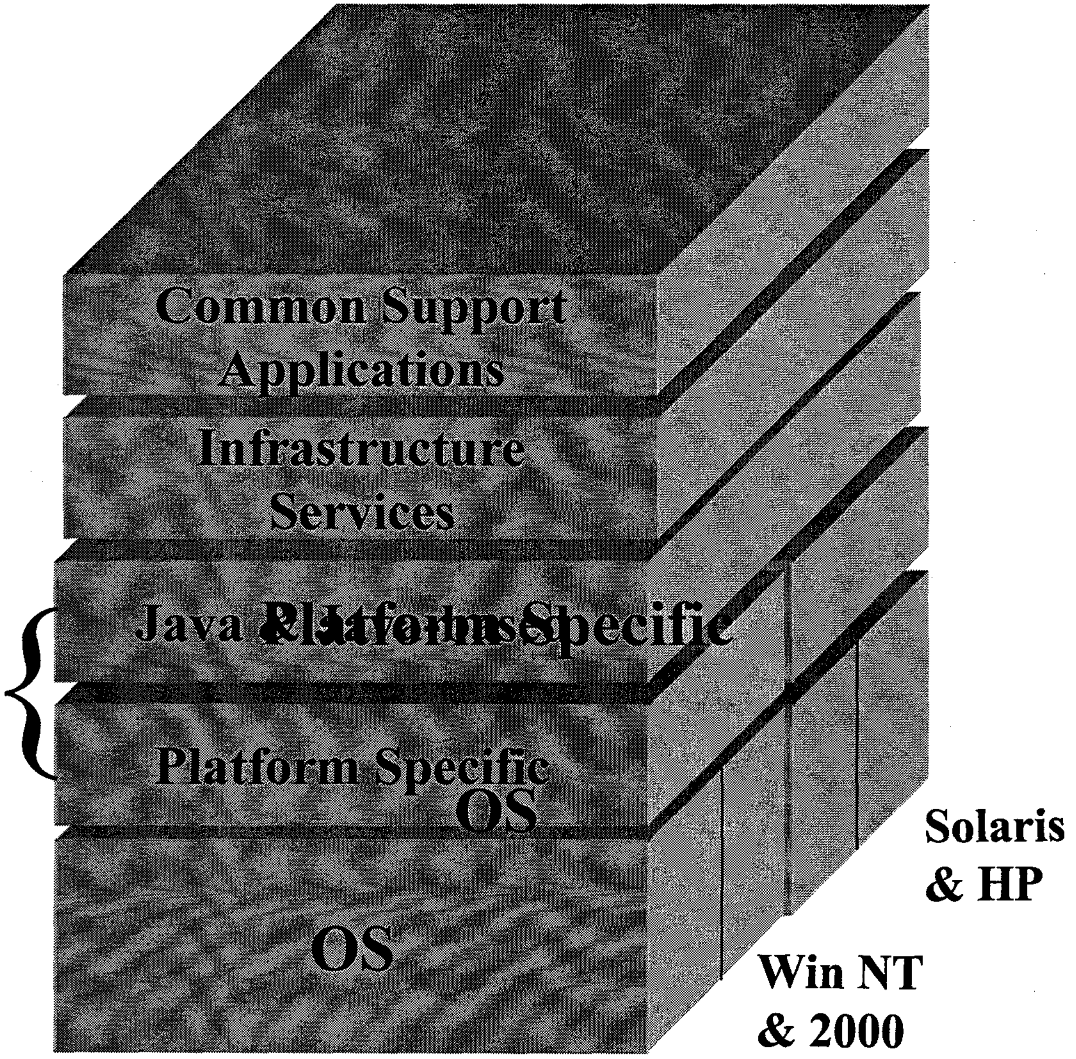
DII COE Architecture

- Kernel Services
  - Sits atop the OS and its related APIs including POSIX
  - Tends to be vertical components with small private platform isolation layers
  - Leverages off of the POSIX standard, when possible



Note: To date, most kernel functions have been inherited, future work Will rely on POSIX as much as possible.

Recent DII COE Architecture (3.4 and 4.x)



Kernel Services
(Our goal is a single layer across platforms)

Realtime Kernel Services V1.0.0.0

- Kernel APIs to support realtime environment provided as initial version in 2nd Qtr FY 01.
- Realtime API Library Function Calls
  - Absolute Local and Zulu Times
    - Common Utility Functions
    - Common Test Utility Functions
    - coeSetTime
    - coeGetTime
    - coeDisplayLocalTime
    - coeDisplayLocalTimeWithoutZone
    - coeDisplayZuluTime
  - Shutdown
    - csaShutdown

Realtime Kernel Services V1.0.0.0

- **Realtime API Library Function Calls**
  (continued)
  - Process/Thread Management
    - cpmSignal
    - cpmKillNamedProcess
    - cpmKillNamedProcesses
    - cpmKillOwnedByName

    cpmKillProcess

    cpmKillProcesses

    cpmKillOwnedByPID

    cpmStartBootProcesses

- **Realtime API Library CLIs**
  - coeDisplayTime
  - cpmKillProcess
  - cpmKillNamedProcess
  - cpmKillOwnedByName
  - csaShutdown

  coeSetTime

  cpmKillProcesses

  cpmKillOwnedByPID

  cpmStartBootProcesses

Some of the Growing Pains

- Inclusion of additional desktop platforms
  - Don’t always support POSIX
  - Often use very different approaches to system services
- Desire to provide support for additional COTS components
  - Need to ensure that business rules are the same
  - Results are interoperable
- Need to support a wide range of installations
  - Single standalone station in the field
  - Distributed systems with 10,000+ users

Some of the Growing Pains

- Emerging need to support realtime and embedded systems as well as co-existent systems (RT and non-RT cohabit)
  - Partial support for PSE54
  - Questionable need for other services (e.g. account management, software installation)
  - Requires the adoption of a different programming model
    - Finer timing resolution
    - Awareness of memory allocation/deallocation
    - Awareness of timeliness, etc.

DEFENSE INFORMATION SYSTEMS AGENCY

JPL

Future Kernel Directions

How can the Open Group Help?

- Changes to APIs
  - Absolute local and Zulu times
  - Shutdown/restart - what's missing?
- Processes/Threads
  - Process management
  - Boot process management (including sequencing)
  - Process monitoring
- Accounts/Profiles
  - Standard APIs for user/group admin
    - Problems with respect to maximum number of users in a group

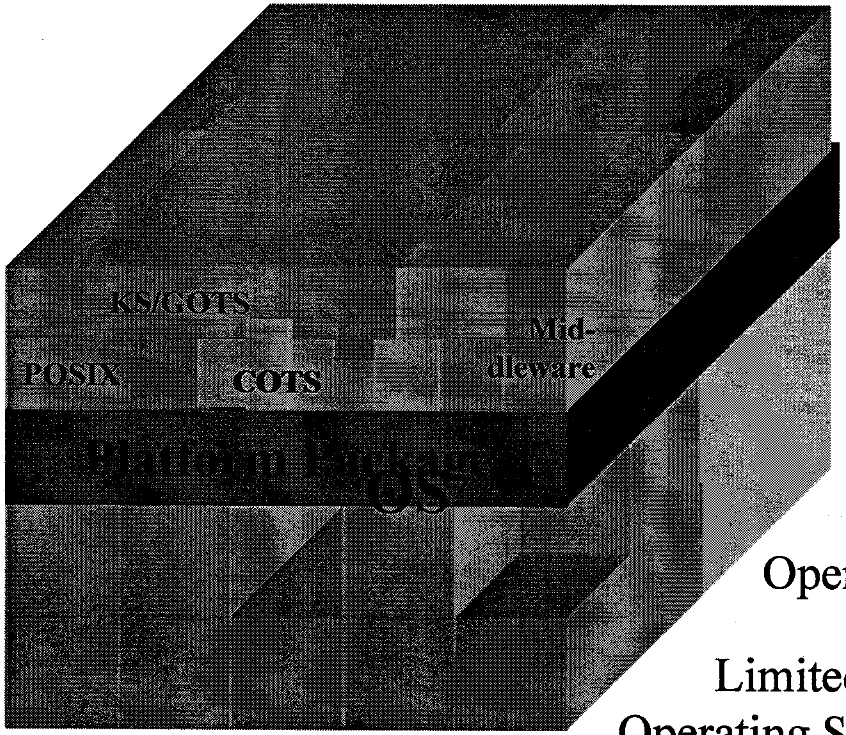
How can the Open Group Help?

- Security Infrastructure (e.g. cross platform access control lists)
- Establish criteria for vendor POSIX compliance
  - Mechanism for compliance reporting
  - Possible compliance test suite creation

Additional Areas of Review

- Establish fundamentals for co-existent systems (combination of non-RT and soft RT components)
  - Defining the boundaries and their interfaces between supported and unsupported DII COE components
    - Disparate time domains (non-RT, soft-RT, hard-RT)
    - Application space (C2, Embedded, etc.)
  - Using full versus limited featured OS
    - PSE54 versus PSE52 et al.
    - Configurability of components
  - Tools for the development and integration environment

Future DII COE Kernel Architecture?



Kernel Services

Operating System

Robust
Operating System

Limited
Operating System
(e.g. O/S without hierarchical file system support)